Keeping Information Tags Valid and Consistent

Karol BALKO*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
balko.karol@gmail.com

Metadata have become inseparable part of modern system development. Metadata as structured information describe the information resources. And so metadata are used for describing the features of the source code in our field of research. The features of the source code include for example the number of white characters or denomination of the copied code.

Our research field is focused especially on the metadata used within the project PerConIK. Within this project metadata are being denominated by the expression information tags. Information tags are stored in the repository of information tags and are connected to the source code through a reference to the source code. However, these information tags can be invalidated during the modification and refactoring of the source code as their references to source code can be invalidated or they can describe invalid feature of the source code.

The thesis aims to create a method to control the consistency and validity of theses information tags. Through the analysis of this problem we have proposed possible method, which will allow us make this control. We used several ways in our solution. In general, as well as the analysis of metadata through approaches used in related fields of research have been analysed.

In our method we use abstraction because the source code is very specific and a transformation of source code to the abstract syntax trees makes the method language independent. One method, which we are trying to use is to examine the abstract syntax trees based on their similarities. To find the similarity we use a method called robust tree edit distance, which shows good performance in temporal and spatial complexity. On the basis of the resulting distance against other abstract syntax trees, we are trying to cluster these abstract syntax trees, for a faster comparison of the new source code that come into the system.

If the evaluation proved our method, we would be able to transfer information tags to source code which is new in system, based on the abstract syntax trees similarity. Although we can't claim certainty that these information tags are absolutely

* Supervisor: Karol Rástočný, Institute of Informatics and Software Engineering

Spring 2014 PeWe Workshop, March 21, 2014, pp. 63–64.

valid, but we can determine their similarity to the nearest cluster of abstract syntax trees.

As clustering algorithm, we consider k-means clustering algorithm first, but the problem is adding new abstract syntax trees without re-clustering all known abstract syntax trees again. So in the next progress of our research, we want to use Birch algorithm, which has a lot of advantages against k-means clustering algorithm. Like incremental adding new abstract syntax trees.

Acknowledgement. This contribution is the partial result of the Research & Development Operational Programme for the project Research of methods for acquisition, analysis and personalized conveying of information and knowledge, ITMS 26240220039, co-funded by the ERDF.

References

[1] R. Koschke, R. Falke, and P. Frenzel, "Clone Detection Using Abstract Syntax Suffix Trees," in Reverse Engineering, 2006. WCRE'06. 13th Working Conference on, 2006, pp. 253 – 262.

[2] M. Pawlik and N. Augsten, "RTED: A Robust Algorithm for the Tree Edit Distance," in The 38th International Conference on Very Large Data Bases, 2011, pp. 334–345.

[3] W. Yang, "Identifying Syntactic Differences Between Two Programs," vol. 21, no. JULY, pp. 739–755, 1991.